

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No. : 10/519,394
Applicant : SIEGELIN, Christoph
Filing Date : 12/22/2004
Confirmation No. : 2863
Art Unit : 2186
Examiner : CHRZANOWSKI, Matthew R.
Docket No. : 76.0733PR
Customer No. : 41754

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF ELECTRONIC TRANSMISSION UNDER 37 CFR 1.8

Date of Transmission: September 13, 2010

I hereby certify that this correspondence is being filed with the United States Patent and Trademark Office using the Electronic Filing System on the date indicated above.

/Pehr Jansson/
Pehr Jansson, Reg. No. 35,759

APPELLANT'S BRIEF1. Real Party in Interest

The real party in interest in this appeal is Gemalto N.V., a corporation of the Netherlands. Axalto SA was named as the assignee when the 35 USC 371 application was filed. Axalto, SA. is a French subsidiary of Gemalto, N.V.

2. Related Appeals and Interferences

There are no related appeals and interferences.

3. Status of Claims

Claims 1 – 23 and 25 – 41 are pending in the application and stand rejected in the Office Action of 16 March 2010 (“Office Action”).

Claims 1 – 23 and 25 – 41 stand rejected.

Claims 11 – 13, 23 – 31, and 38 – 41 are cancelled (per amendment after final filed on July 7, 2010).

Claims 1 – 10, 14 – 22, and 32 – 33 are appealed herein.

Claims 1 – 10, 14 – 22, and 32 – 33 stand rejected solely under 35 USC 102 or 35 USC 103. The rejections to Claims 1 – 10, 14 – 22, and 32 – 33 are appealed herein.

Claims 11 – 13, 23 – 31, and 38 – 41 stand rejected under either 35 USC 112 first paragraph or 35 USC 101. Applicants, without agreeing to the rejections of 11 – 13, 23, 25 – 31, and 38 – 41 cancelled those claims in the amendment filed on July 7, 2010.

4. Status of Amendments

Appellants attempted to cancel Claims 11 – 13, 23 – 31, and 38 – 41 in an amendment filed on July 7, 2010. No advisory action has been issued to date.

Appellants believe that these cancellations clarify the issues for appeal in removing non-art rejections under 35 USC 101 and 35 USC 112, first paragraph. Appellants discussed the proposed cancellations with the Examiner on September 10, 2010 and the Examiner indicated that the cancellations would moot the 35 USC 101 and 35 USC 112 rejections.

Appellants co-file herewith a Third Amendment After Final. Appellants have discovered a minor omission from Figure 1, namely, element 2 (electronic unit) was inadvertently left out of the figure. The Third Amendment After Final proposes a correction to Figure 1 in the form of a replacement page.

5. Summary of Claimed Subject Matter

Claim 1 is directed to a method to write in flash type memory (Spec Page 1, Lines 2 – 3; Figure 1: 7) of an electronic module (Specification, Page 4, Lines 10 – 20; Figure 1: 2) comprising

defining a mirror area (Specification, Page 4, Lines 24 – 26; Spec, Page Figure 2: the contiguous memory area including the n areas ZP1, ZP2 through ZPN; Figures 3, 4: the four contiguous memory areas ZPu and ZPi) in the flash type memory (Figure 1, 7) divided into at least two physical areas (Specification, Page 4, lines 24 – 25; Figure 2: ZP1, ZP2, ... ZPN) each designated to correspond to a same logical area (Specification, Page 4, Line 25; Figures 2, 3, 4: ZL; Figure 5) for storing content to be written to the logical area (Specification, Page 4, Lines 29 – 30; Figure 2, ZL);

designating one of the at least two physical areas as being an active physical area (Specification, Page 4, Line 28 – Page 5, Line 4; Specification, Page 5, Line 30 – Page 6, Line 4; Specification, Page 6, Lines 5 - 22; Figure 3: Active ZP RAM Pointer Figure 2: ZP1, ZP2, ... ZPN; Figure 4: e.g., least significant area designated as “used”; Page Figure 3: ZPu area pointed to by the Active ZP RAM Pointer); and

during a write to said logical area, programming the content of said logical area (Specification, Page 6, Line 29 – Page 7, Line 22; Figure 5: the area signified by dashed boundary is the “logical area”, into the active physical area Figure 5a, 5b: ZPn; Figure 5c: ZPn+1).

6. Grounds for Rejection to be Reviewed on Appeal¹

35 USC 102(b)

¹ Appellants have elected to not appeal the rejections under 35 USC 101 and 35 USC 112, first paragraph.

Claims 1 – 2 and 32 – 33² stand rejected under 35 USC 102(b) as anticipated by Ban (WO 94/20906, hereinafter “Ban”).

35 USC 103(a)

Claims 3, 7 – 8, and 18³ stand rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view of Assar et al. (WO 95/10083, hereinafter “Assar”).

Claim 4⁴ stands rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view of Mennecart (WO 95/10083, hereinafter “Mennecart”).

Claim 5⁵ stands rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view of Hazen (WO 99/35650, hereinafter “Hazen”).

Claim 6⁶ stands rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view Hazen and in view of Lipoviski (US Patent #5758148, hereinafter “Lipovski”).

² Claims 11, 13, 23, 34 – 35, and 38 – 41 were also rejected under 35 USC 102(b) as anticipated by Ban. However, because Appellants is not appealing the non-art grounds for rejection of these claims, Appellants, while not agreeing that Ban anticipates these claims, are not appealing their rejection.

³ Claims 28 and 29 were also rejected under 35 USC 103(a) over the combination of Ban and Assar. However, because Appellants is not appealing the non-art grounds for rejection of these claims, Appellants, while not agreeing that these claims are unobvious over Ban and Assar, Appellants are not appealing their rejection.

⁴ Claim 25 was also rejected under 35 USC 103(a) over the combination of Ban and Mennecart. However, because Appellants is not appealing the non-art grounds for rejection of these claims, Appellants, while not agreeing that this claims are unobvious over Ban and Mennecart, Appellants are not appealing its rejection.

⁵ Claim 26 was also rejected under 35 USC 103(a) over the combination of Ban and Hazen. However, because Appellants is not appealing the non-art grounds for rejection of these claims, Appellants, while not agreeing that this claims are unobvious over Ban and Hazen, Appellants are not appealing its rejection.

⁶ Claim 27 was also rejected under 35 USC 103(a) over the combination of Ban, Hazen and Lipovski. However, because Appellants is not appealing the non-art grounds for rejection of this claim, Appellants, while not agreeing that this claims are unobvious over Ban, Hazen, and Lipovski, Appellants are not appealing its rejection.

Claims 9 – 10⁷ stands rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view of Kuo (US Patnt # 4763305, hereinafter “Kuo”).

Claims 14 – 17 stands rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view Hazen and in view of Assar.

Claims 19 – 20 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view Hazen and in view of Kuo.

Claims 21 – 22 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Ban in view Assar and in view of Kuo.

7. Argument

The Problem

Appellants deal with the problem of efficiently writing into a flash memory.

According to the Specification, a flash memory is defined as a non-volatile memory with a dissymmetry between the time required for programming and erasure. Specification, Page 3, Line 31 – Page 4, Line 1. Three operations may be defined for flash memories: erasure, programming, and writing. Erasure consists of switching an entire region, referred to as a *block*, to “low” state, i.e., turning the entire block to logic 0. Programming consists of switching specific bits from to “high” state, i.e., logic 1.

⁷ Claims 30 and 31 were also rejected under 35 USC 103(a) over the combination of Ban and Kuo. However, because Appellants is not appealing the non-art grounds for rejection of these claims, Appellants, while not agreeing that these claims are unobvious over Ban and Kuo, Appellants are not appealing their rejection.

Writing consists of erasure of a block (i.e., setting the entire block to logic 0) and programming selected bits (i.e., setting those selected bits to logic 1). Specification, Page 1, Lines 18 – 24.

When a write operation would turn at least one bit from logic 1 to logic 0, an entire block would have to be erased. A problem with write operations is that erasure is very slow and many flash memories have a limited number of erase cycles available for each block.

It is therefore, desirable to have a solution that allows for writing logic 0 to bits previously programmed as logic 1 without requiring block erasure or, more precisely, reducing the number of erasures and deferring such erasures until a convenient time. Appellants propose and claim such a solution.

A possible solution

Before discussing Appellants' solution, let's consider one possible solution. Consider physical memory as a contiguous address space divided into blocks. Consider that computer programs write into a dissociated address space referred to as logical address space. When a program accesses a particular memory block, the program addresses that block using the logical address. A table provides a mapping between the logical block and the corresponding physical block. A translation step occurs at the system level when the program addresses the logical block to provide access to the correct physical block corresponding to the addressed logical block.

When a write operation occurs that requires writing a logic 0 into a location previously programmed as logic 1, if written directly into the physical block, the entire physical block would have to be block erased and re-programmed, except for the bit that is now bear the value logic 0. However, in one possible solution, an unused block having all values as logic 0 could be found, and the translation table remapped so that the logical address to the logical block would be mapped to the unused block (the new physical block), and the new physical block could be programmed to have the values of the logical block.

Thus, in this solution, a pool of unused blocks (for example, all the unused blocks in the memory) are made available to be remapped to correspond to logical blocks so that when a block has to be written to, the erase operation is avoided.

Naturally, periodically when the pool of unused blocks have been depleted, old blocks have to be reclaimed and some provisions may have to be made to avoid problems with incomplete writing of blocks due to tearing (the unexpected shut-down of the memory during a write operation).

Appellants' solution

Appellants disclose and claim an alternative to the solution described above. Consider Appellants' Figure 2:

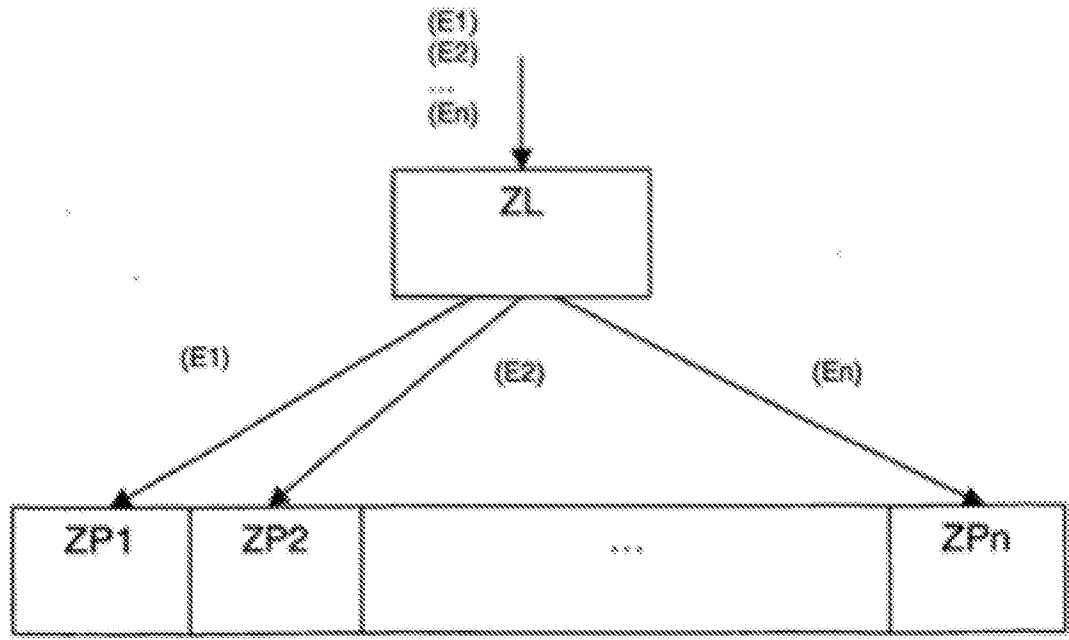
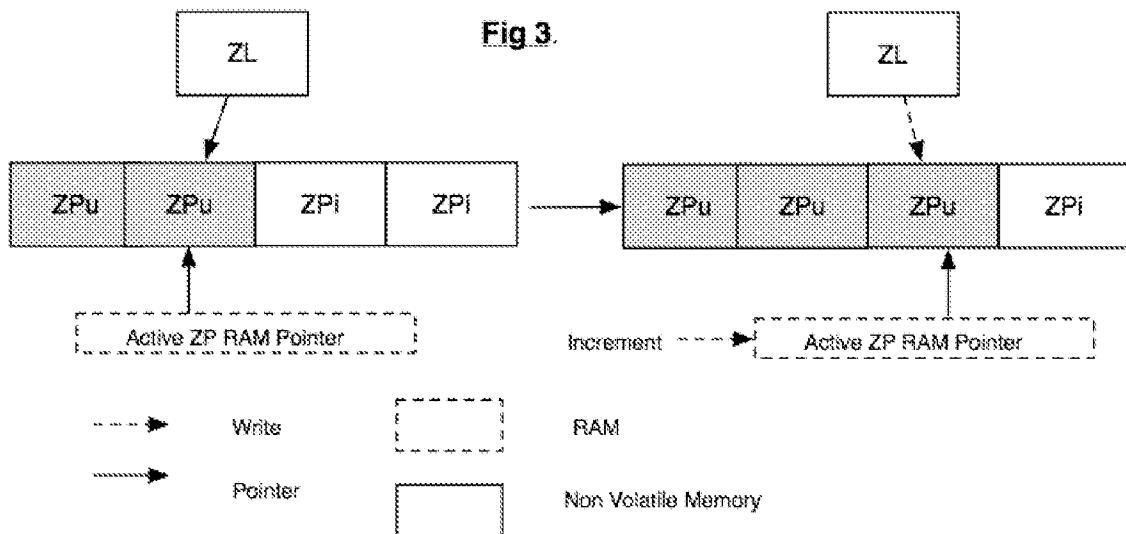


FIG.2

The element depicted as ZL is a logical area. The elements depicted ZP1, ZP2, ... ZPN are areas in the physical memory and together form a “mirror” memory in which each of the areas ZP1 ... ZPN correspond to the same logical area ZL. The steps (E1), (E2) ... (EN) adjacent to the vertical down arrow at the top of the figure illustrate write operations from a program.

When a program writes, it writes to a logical area that is translated into a physical area in the mirror area referred to as the “active” area.

Consider now Figure 3 of Appellants’ application:



Used areas in the mirror memory corresponding to **ZL** are shaded and bear the label **ZPu** (u for used) and the not-yet used areas are not shaded and have the label **ZPi**. In the left-hand portion of the figure, an “Active ZP RAM Pointer” points to the least-significant of the used areas thereby directed the logical area **ZL** to access that area. Now consider the write operation of the right-hand side of the figure. The “Active ZP RAM Pointer” is incremented to point to the next-least significant area in the mirror area and the write operation proceeds into that area.

The claims

The above described technology is claimed by Appellants as follows:

1. A method to write in flash type memory of an electronic module comprising:

defining a mirror area in the flash type memory divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area;
designating one of the at least two physical areas as being an active physical area; and
during a write to said logical area, programming the content of said logical area into the active physical area.

Thus, the claimed invention is directed to a method for writing into a flash type memory. The method comprises a first step of defining a mirror area (i.e., the combined area of areas ZP1, ZP2, ... ZPN) divided into at least two physical areas (e.g., ZP1, ZP2, etc.) that correspond to a same logical area (ZL) for storing content written to the logical area. One of the areas is designated as the active area (e.g., the “Active ZP RAM Pointer” of Figure 3) and during a write, programming the content into the active physical area.

Ban

The Examiner rejected Claim 1 as anticipated by Ban. Appellants traverse this rejection and respectfully requests its reversal.

Ban also deals with writing to a flash memory (Ban, Page 1, Lines 4 – 6) and makes some of the same observations discussed herein above, e.g., “In a flash memory, it is not practical to rewrite a previously written area of the memory without a preceding block erase of the area” (Ban, Page 1, Lines 14 – 16) and “Thus, flash memories are not compatible with typical existing operating system programs, since data cannot be written

to an area of flash memory in which data has previously been written, unless the area is first erased” (Ban, Page 1, Lines 26 – 29).

Ban does provide a mechanism for avoiding erasing memory blocks on writes. However, Ban proposes a very different mechanism from the one claimed by Appellants; one that is similar to the solution described above under the heading “one possible solution.”

“In a write operation, … an unwritten block is therefore located and written to.” Ban, Page 3, Line 21.

Thus, in the context of the present application, it is interesting to investigate how Ban locates an unwritten block during write operations and to contrast that against the claimed mechanism.

To understand Ban it is useful to understand the addressing structure proposed therein. Ban provides a two-level addressing scheme in which virtual addresses correspond to logical addresses which in turn correspond to physical addresses. Ban, Page 8, Lines 20 – 24. Ban provides for “the provision of a flash memory, virtual mapping system that allows data to be continuously written to unwritten physical address locations. The virtual memory map relates flash memory physical location addressees in order to track the location of data in the memory.” Ban, Page 2, Lines 16 – 20.

“The virtual map is a table in which the first entry belongs to virtual block 0, the second to virtual block 1, and so on. Associated in the table with each virtual block address there is a corresponding physical address. … The virtual memory map is used to convert the virtual block address to a physical block address.” Ban, page 3, Lines 8 – 14.

Each of Ban's addresses has a block number and a block offset. Figure 4, element 29; Ban, Page 7, lines 17 – 19. A virtual map 31 maps block numbers to logical unit address. Ban, Page 7, Line 30. The logical unit address includes the logical unit number. Ban, page 8, Line 6. The logical unit table 35 translates the logical unit number to a physical unit number for the logical unit. Ban, Page 8, Lines 8 – 9.

Consider Figure 4 of Ban:

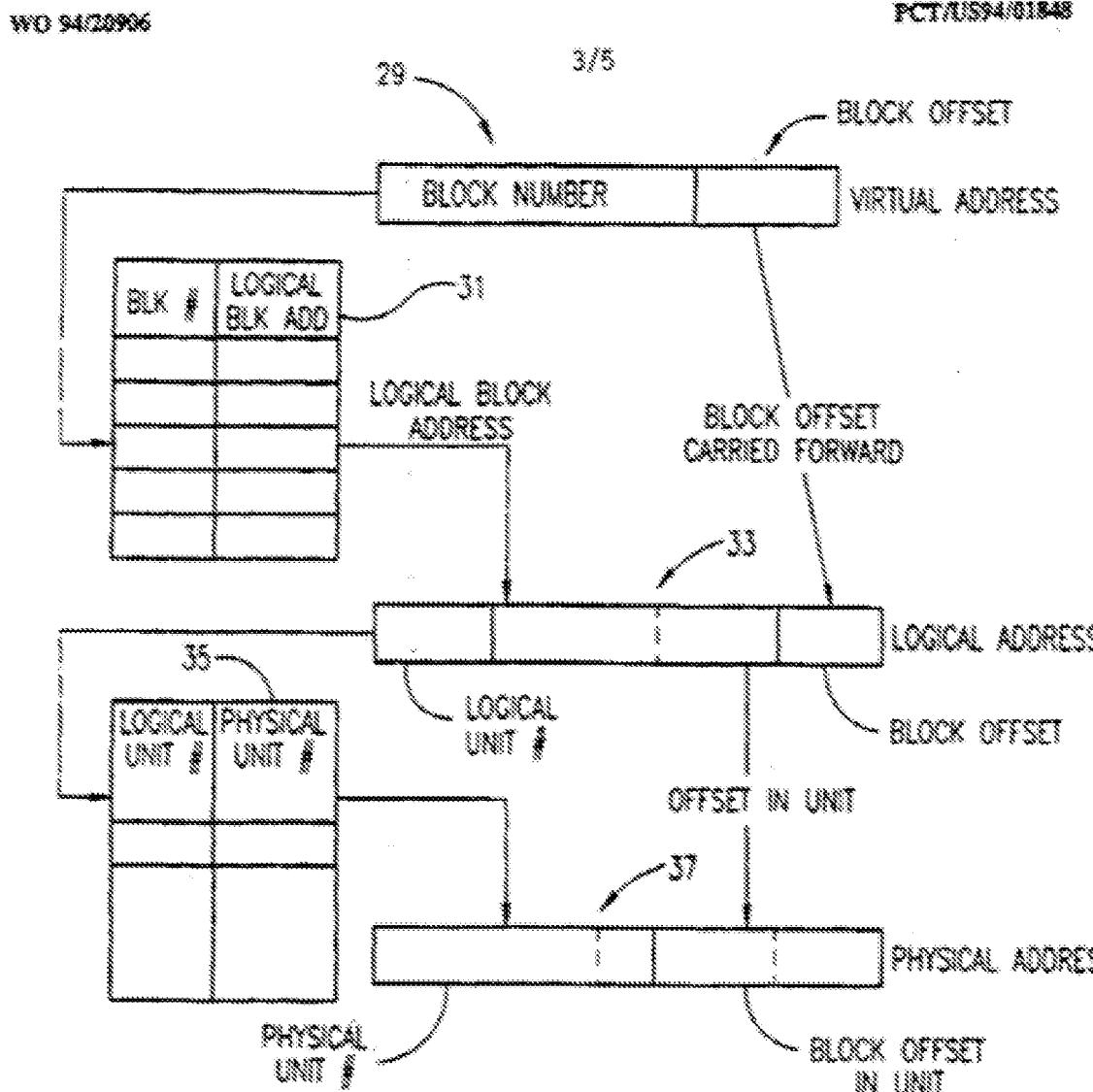


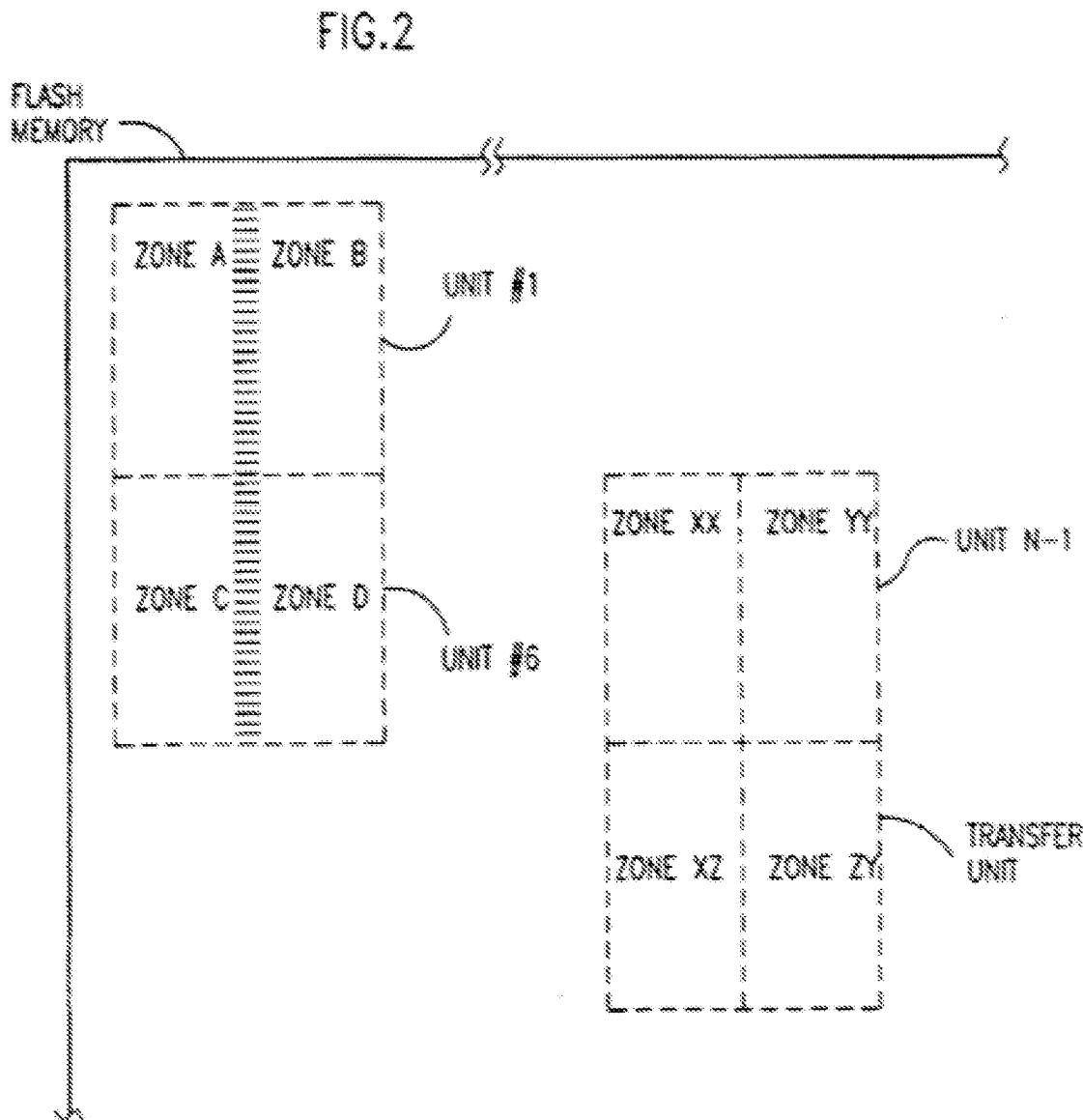
FIG.4

It is quite evident how the block number is translated first into a logical address from table 31 and then the logical unit extracted from the logical address and mapped to a physical unit.

Now let's consider what Ban considers as units and blocks. Ban's memory organization starts with zones. Page 6, Lines 14 – 25. "Each zone is comprised of a

number of contiguous physical memory locations that can be block erased using conventional, well known, flash memory technology.” Lines 18 – 20. The zones are organized as units of a plurality of contiguous zones. Lines 22 - 25. “Each unit is comprised of an integral (sic – we assume Ban means integer) number of addressable blocks.”

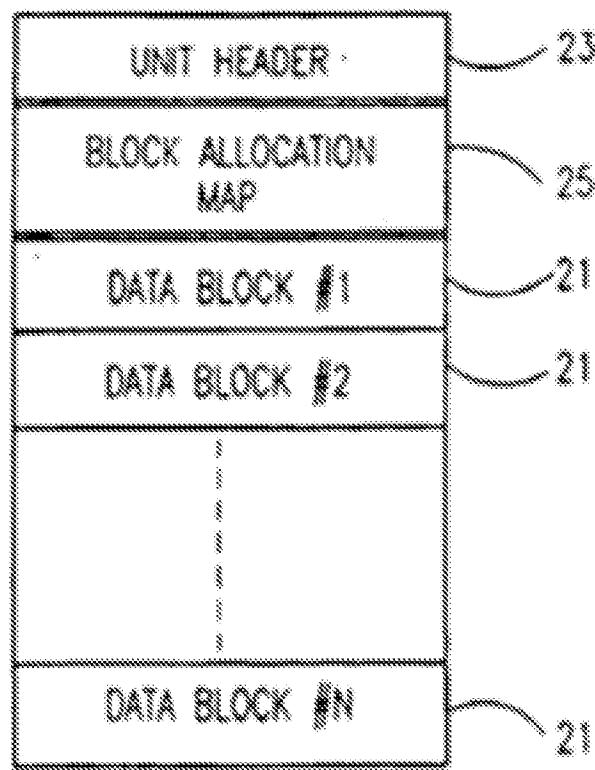
Ban’s Figure 2 illustrates the relationship between zones and units:



Thus, it can be seen that each unit comprises multiple zones.

Figure 3 illustrates the relationship between units and blocks (“Referring now to Figure 3, each unit contains an integral [integer?] number of contiguous data blocks 21” (Ban, Page 7, Lines 1 - 2)) :

FIG.3



Turning now to Ban’s way of addressing write operations. “In a write operation, the virtual address 29 [i.e., the address from a computer program] is again mapped initially to a logic unit number and a block offset with the unit.” Ban, Page 8, Lines 26 – 27. “If the block corresponding to this address has been written, a write command cannot be executed at the corresponding physical address.” Ban, page 8, Lines 28 – 30. Thus, the problem of writes has been recognized. To address the problem, “[the] control algorithm scans the block allocation maps 25 for each unit until a free block is located.”

“The block allocation map 25 has a word for each block that denotes its status [as] ‘block free and writable’; ‘block deleted and now writable’; ‘block allocates and contains user data.’” Ban, Page 7, Lines 11 – 14. Thus, the control algorithm looks to multiple units (“for each unit”) looking for a free block. “The virtual map 31 is update[d] so that the original virtual address now points to the new logical address where the write operation is to take place. This logical address is mapped to a physical address, in the manner previously described, and the block is written to this address.” Ban, Page 9, Lines 4 – 8.

From the foregoing, it is clear that what Ban describes is that when a block is to be written to, all blocks of all units are candidates to become the new block to which the data is to be written. That is manifestly different from Appellants’ claim which prescribes “defining a mirror area … divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area.” In Ban, the blocks (which is the unit that corresponds most closely to the “areas” of Appellants’ claims) are in no way designated to correspond to a same logical area. Rather Ban teaches away from this limitation by explicitly stating that all units are scanned to find an available free block.

Appellants further claim “designating one of the at least two physical areas as being an active physical area.” The *the* of “one of *the* at least two physical areas” of course specifies that the *one* of the two physical areas is one of the two physical areas of the defined mirror area. As the designated area for the free block located in Ban by scanning the block allocations maps for each unit, the new block (defined by the new

logical address) cannot be considered as “*one of the* at least two physical areas” as it is not found in a mirror area.

Finally, Claim 1 recites “during a write to said logical area, programming the content of said logical area into the active physical area.” This, of course, refers to that the active physical area is one of the areas in the mirror memory. Similarly as with the previous element, in Ban the new block is allocated from any unit (as illustrated by the fact that the control algorithm scans each unit for a free block). Thus, the block that is written to by Ban is not a block in a mirror area because Ban does not use a mirror area to designate a correspondence to logical unit blocks.

The Examiner argues that since each block is formed from a fixed-length group of physical byte addresses and one or more physically contiguous flash memory areas or zones comprise a number of blocks, that that is a teaching of defining a mirror area divided into at least two physical areas designated to correspond to a same logical area because “therefore the physical byte addresses are associated with logical blocks, zones and units.” Office Action, Page 6, Line 21 – Page 7, Line 9. However, all that is is an explanation of the addressing scheme used by Ban. That multiple contiguous byte addresses can be grouped into blocks, zones, and units is hardly new. However, that is not what Appellants claim. Appellants claim defining, in a physical space, areas that are mirror areas that are divided into physical areas that each correspond to the same logical area for storing the same content written to the logical area. In Ban’s addressing scheme, if the analogy is taken as a premise for an argument, the various components that make

up the addressing hierarchy (bytes, blocks, zones, units) are never meant to store the same content as one particular logical area but rather to store portions of a logical area.

For the foregoing reasons it must be clear that Ban does not teach or suggest the mechanism claimed by Appellants in Claim 1. Accordingly, Appellants respectfully request reversal of the rejection.

Claims 2, 32, and 33

Claims 2, 32, and 33 depend from Claim 1, incorporates all the limitations thereof, provide further unique and non-obvious combinations, and are patentable over Ban for the reasons given in support of Claim 1 and by virtue of such further combinations.

Ban in view of Assar

Claims 3, 7 – 8, and 18 stand rejected under 35 USC 103(a) as unpatentable over Ban in view of Assar. Appellants traverse the rejection and respectfully requests its reversal.

Assar, like Ban, fails to teach or suggest defining mirror areas for designating a new area to write into. Assar teaches that “if a data block is changed and stored again, it is placed into a location in the memory storage which is empty.” Assar, Page 20, Lines 6 – 8. That begs the question “how is that empty location determined?” Assar provides a “second algorithm … for leveling erase cycles among all the blocks within the entire

mass storage device.” Assar, Page 11, Lines 3 – 4. “In this way, sections of the mass storage which have been erased numerous times are programmed with a reallocation data file which is rarely changed thereby allowing all sections of the mass storage to eventually approach parity of erase cycles.” Assar, Page 11, Lines 17 – 21. The foregoing implies that allocation of a block for writing a changed data block may draw from anywhere in the mass storage device. Thus, Assar teaches away from “defining a mirror area.” For this reason, at least, Claim 1 would be patentable over Assar and the proposed combination with Ban.

Claims 3, 7 – 8, and 18 depend from Claim 1, incorporate all the limitations thereof, provide further unique and non-obvious combinations and are patentable at least for the reason given above in support of Claim 1 and by virtue of such further combinations.

Claim 3

Claim 3 recites “the convenient time is a period of inactivity or when all the physical areas are used.” It must be noted that “the physical areas” refers back to the physical areas of the defined mirror area.

Assar teaches that “if a data block is changed and stored again, it is placed into a location in the memory storage which is empty.” Assar, Page 20, Lines 6 – 8. That begs the question “how is that empty location determined?” Assar provides a “second algorithm … for leveling erase cycles among all the blocks within the entire mass storage device.” Assar, Page 11, Lines 3 – 4. “In this way, sections of the mass storage which

have been erased numerous times are programmed with a reallocation data file which is rarely changed thereby allowing all sections of the mass storage to eventually approach parity of erase cycles.” Assar, Page 11, Lines 17 – 21. The foregoing implies that allocation of a block for writing a changed data block may draw from anywhere in the mass storage device. Thus, Assar teaches away from “defining a mirror area.”

The Examiner makes the argument that Assar’s teaching of that when the physical memory is filled, blocks with certain flags are erased, wherein as described above blocks contain multiple physical mirror areas.” Office Action, Page 13, Lines 18 – 21, *citing* Assar, Page 20, Lines 10 – 19.

As noted above, Assar does not teach or suggest using mirror areas to designate the new block to write into. In the cited passage of Assar, Assar states that “periodically the memory storage will fill” thus implying that the entire memory storage is full and not merely a mirror area. Thus, Assar does not teach or suggest that any erase operations cannot be triggered by “when all the physical areas [of the mirror area] are used.”

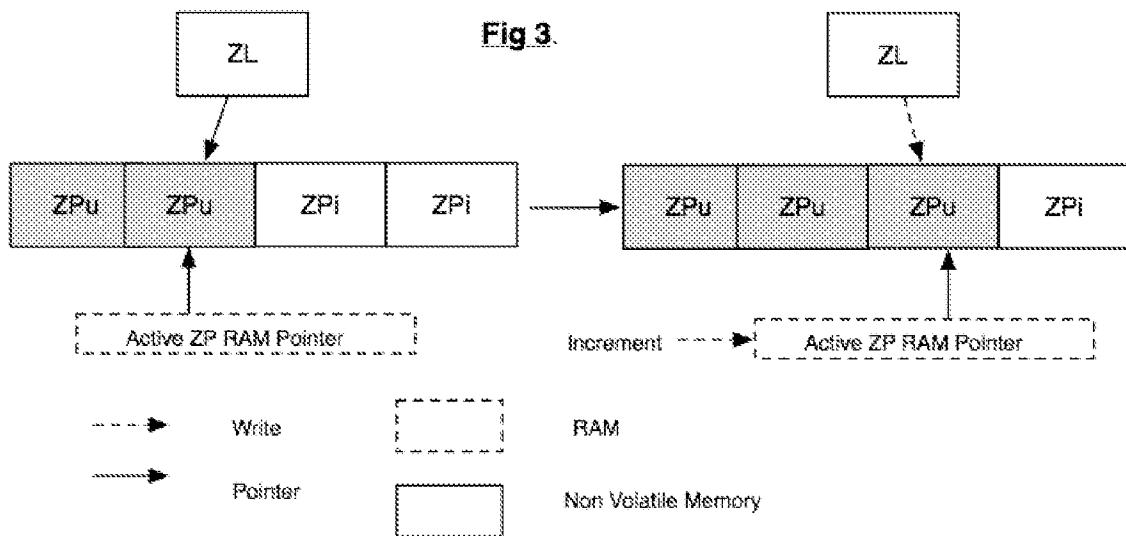
For the foregoing additional reason, Claim 3 is not taught or suggested by the combination of Ban and Assar. Accordingly, Appellants respectfully request withdrawal of the rejection.

Claim 7

Claim 7 recites “designating said active physical areas using a counter and incrementing the counter on each change of active area.” The Examiner cites Assar’s “wear out leveling counter 620” as equivalent to the claimed element (“Assar discloses a

method comprising designating active physical areas using a counter (Counter 620, page 18, liens 26 – page 19, line 1; page 20 lines 17 and 26; counter 620 is used in conjunction with flags such as the used/free flag 112) and incrementing the counter on each change of active area (page 20, lines 10 – 19). Appellants traverse this rejection and respectfully request reversal of it.

Appellants again refer to Figure 3:



As can be seen in the figure, incrementing the counter is used to point to the next physical area in the mirror area. Because the area is “divided” it may be inferred that the blocks are contiguous in the physical address space. It is therefore possible to move up to the next area by incrementing a pointer.

The wear leveling counter 620 is used to determine how many times a given block has been erased (“As part of the erase operation, the wear out leveling count 620 for each erase data block and information block will be incremented.” Assar, Page 20, Lines 16 –

19.). Thus, the counter is not incremented to designate a new block; rather, it is incremented when erased. Erasure is the very act that is avoided during appellants' claimed write operation!

Therefore, modifying Ban to include the wear out leveling counter 620 would not yield a mechanism of "designating said active physical areas using a counter and incrementing the counter on each change of active area."

For this additional reason, Appellant posit that Claim 7 is not unpatentable over the combination of Ban and Assar.

Ban in view of Mennecart

Claim 4 stands rejected as unpatentable over Ban in view of Mennecart. Appellants traverse the rejection and respectfully request reversal thereof.

Mennecart does not describe the allocation of a new block into which to write a content of a logical area. Therefore, it is not surprising that Mennecart, like Ban, fails to teach or suggest defining a mirror area in which one physical area may be designated as the active area and writing into that active area. Accordingly, Claim 1 is patentable over the combination of Ban and Mennecart. Claim 4, depends from Claim 1, inherits all the limitations thereof, provide further unique and non-obvious combinations, and is patentable for at least the reasons given in support of Claim 1, and by virtue of such further combinations.

Claim 4 adds the additional limitations of “copying the active physical area into a buffer area, erasing all physical areas and copying the buffer into a first available physical area in the mirror area.” Of course, “*the active physical area*” refers to a physical area in the mirror area. Because Mennecart fail to teach or suggest the use of a mirror area from which to allocate an active area into which the contents of the logical area is to be written, Mennecart fails to teach or suggest “copying the active physical area [in the mirror area] into a buffer … and copying the buffer into a first available physical area in the mirror area.”

Furthermore, Mennecart does not teach allocating a new physical area for writing into. Rather, Mennecart teaches buffering incoming data and erasing *the zone* into which that data is to be written. Thus, Mennecart does not teach “erasing all physical areas … in the mirror area” (Claim 4), merely erasing *the one* area into which the data is to be written.

For the foregoing additional reasons, Claim 4 is patentable over the suggested combination of Ban and Mennecort and Appellants respectfully request reversal of the rejection of Claim 4.

Ban in view of Hazen

Claim 5 stands rejected as unpatentable over Ban in view of Hazen. Appellants traverse the rejection and respectfully request reversal thereof.

Hazen does not deal with avoiding the problem of having to erase a block of flash memory in order to write into the block. It is not surprising then that Hazen does not

describe allocating a new block for making writes into and, in particular, that it does not describe using a mirror area divided into physical areas for storing the content of a logical area. Therefore, the combination of Ban and Hazen would lack the elements argued hereinabove as missing from Ban and Claim 1 is patentable over the suggested combination.

Claim 4, depends from Claim 1, inherits all the limitations thereof, provide further unique and non-obvious combinations, and is patentable for at least the reasons given in support of Claim 1, and by virtue of such further combinations.

Ban in view of Hazen in view of Lipovski

Claim 6 stands rejected as unpatentable over Ban in view of Hazen in view of Lipovski. Appellants traverse the rejection and respectfully request reversal thereof.

As noted above, Ban and Hazen both fail to teach or suggest “defining a mirror area in the flash type memory divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area; designating one of the at least two physical areas as being an active physical area; and during a write to said logical area, programming the content of said logical area into the active physical area.” Claim 1.

Lipovski does not deal with flash memories. Lipovski, like Ban and Hazen, fail to teach or suggest these elements. That is not surprising because Lipovski deals with writing into dynamic RAM memories which do not suffer from the requirements of flash memories in which block erases are necessary to write.

Therefore, the suggested combination of Ban, Hazen and Lipovski would fail to include the limitations of Claim 1.

Furthermore, Lipovski does not deal with memories in which an entire block must be block erased prior to writing to that block. Therefore, a person skilled in the art would not be motivated to combine Lipovski with Ban or Hazen because Lipovski does not provide solutions for the problem of how to avoid the problems associated with writing into a flash memory.

Claim 6, depends from Claim 1, inherits all the limitations thereof, provide further unique and non-obvious combinations, and is patentable for at least the reasons given in support of Claim 1, and by virtue of such further combinations.

Ban in view of Kuo

Claims 9 – 10 stand rejected under 35 USC 103(a) as unpatentable over Ban in view of Kuo. Appellants traverse the rejection and respectfully requests its reversal.

Kuo, while dealing with avoiding unnecessary erases to an EEPROM, does not discuss designating a new area into which writes are to be made to avoid erasing a block. Therefore, Kuo, like Ban, does not provide a mechanism that includes “defining a mirror area in the flash type memory divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area; designating one of the at least two physical areas as being an active physical area; and during a write to said logical area, programming the content of said logical area into the active physical area.” Claim 1.

Therefore, the suggested combination of Ban and Kuo would fail to include the limitations of Claim 1.

Claim 9 and 10, depend from Claim 1, inherit all the limitations thereof, provide further unique and non-obvious combinations, and are patentable for at least the reasons given in support of Claim 1, and by virtue of such further combinations.

Other combinations.

Claims 14 – 17, 19 – 20, and 21 – 22 stand rejected as unpatentable over various combinations of Ban, Hazen, Assar, and Kuo. As discussed herein above, all these references fail to teach or suggest “defining a mirror area in the flash type memory divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area; designating one of the at least two physical areas as being an active physical area; and during a write to said logical area, programming the content of said logical area into the active physical area.” Claim 1.

Therefore, the suggested combinations of Ban, Hazen, Assar and Kuo would fail to include the limitations of Claim 1.

Claim 14 – 17, 19 – 20, and 21 – 22, depend from Claim 1, inherit all the limitations thereof, provide further unique and non-obvious combinations, and are patentable for at least the reasons given in support of Claim 1, and by virtue of such further combinations.

Conclusion of Argument

Appellants have argued hereinabove that the rejections under 35 USC 103(a) are improper and that the claims are patentable over the prior art. Accordingly, Appellants respectfully request reversal of the rejections of Claims 1 - 6 and their early allowance.

Respectfully Submitted,

/Pehr Jansson/

Pehr Jansson
Reg. No. 35,759

Date: September 13, 2010

The Jansson Firm
9501 N. Capital of TX Hwy.
Austin, TX 78759
512-372-8440 x 200
512-597-0639 (fax)
pehr@thejanssonfirm.com

8. Claims appendix

Listing of the claims:

1. A method to write in flash type memory of an electronic module comprising:
 - defining a mirror area in the flash type memory divided into at least two physical areas each designated to correspond to a same logical area for storing content written to the logical area;
 - designating one of the at least two physical areas as being an active physical area; and
 - during a write to said logical area, programming the content of said logical area into the active physical area.
2. The method according to claim 1, further comprising:
 - erasing the content of all physical areas in a memory area in a single operation at a convenient time.
3. The method according to claim 2, wherein the convenient time is a period of inactivity or when all the physical areas are used.
4. The method according to claim 1, comprising copying the active physical area into a buffer area, erasing all physical areas and copying the buffer into a first available physical area in the mirror area.
5. The method according to claim 2 comprising performing the erasure and programming/read operations in parallel thereby not blocking the electronic module.
6. The method according to claim 5, comprising:
 - performing the erasure and programming/read operations in parallel in a bi-bank memory, said bi-bank memory corresponding to the mirror memory area each bank having physical area(s), one bank being used for programming/reading while the other bank is erased,

- changing active bank when all physical areas of the bank used for programming/read have been used.
7. The method according to claim 1 comprising designating said active physical areas using a counter and incrementing the counter on each change of active area.
 8. The method according to claim 1 comprising associating at least one bit with a logical area to represent the use state of at least one physical area of said logical area.
 9. The method according to claim 1 wherein if the content of the logical area is identical to the content of the active physical area or when said write involves no erasure, the write is carried out in an active physical area and in a blank physical area in the mirror area otherwise.
 10. The method according to claim 9, comprising programming only a portion of the logical area in the blank physical area.
 14. The method according to claim 5, comprising designating said active physical areas using a counter and incrementing the counter on each change of active area.
 15. The method according to claim 6, comprising designating said active physical areas using a counter and incrementing the counter on each change of active area.
 16. The method according to claim 5, comprising associating at least one bit with a logical area to represent the use state of at least one physical area of said logical area.
 17. The method according to claim 6, comprising associating at least one bit with a logical area to represent the use state of at least one physical area of said logical area.

18. The method according to claim 7, comprising associating at least one bit with a logical area to represent the use state of at least one physical area of said logical area.
19. The method according to claim 5, wherein if the content of the logical area is identical to the content of the active physical area or when said write involves no erasure, the write is carried out in an active physical area and in a blank physical area otherwise.
20. The method according to claim 6, wherein if the content of the logical area is identical to the content of the active physical area or when said write involves no erasure, the write is carried out in an active physical area and in a blank physical area otherwise.
21. The method according to claim 7, wherein if the content of the logical area is identical to the content of the active physical area or when said write involves no erasure, the write is carried out in an active physical area and in a blank physical area otherwise.
22. The method according to claim 21, comprising programming only a portion of the logical area in the blank physical area.
32. The method of claim 1, wherein each physical area has a status which is one of three statuses: blank, active and used.
33. The method of claim 32, wherein:
 - the blank status corresponds to one of the physical areas ready to receive data but not selected for receiving data,
 - the active status corresponds to one of the physical areas ready to receive data and selected for receiving data or to one of the physical areas containing the actual content of the logical area to be read,
 - the used status corresponds to one of the physical areas containing an outdated data that shall not be read, said physical area waiting for an erasure.

9. Evidence appendix

Not Applicable.

10. Related proceedings appendix

Not applicable.